**SQL Data Import Project Report: Detailed Workflow, Errors, and Resolutions**

**1. Project Goal**

To import CSV sales data into SQL Server with correct data types, handle NULL/blank values properly, and maintain data integrity in a table dbo.Retail\_sales.

**2. Table Schema (Final Desired Format)**

CREATE TABLE dbo.Retail\_sales (

transactions\_id VARCHAR(50) NOT NULL,

sales\_date DATE NULL,

sales\_time TIME(7) NULL,

customer\_id INT NULL,

gender VARCHAR(10) NULL,

age INT NULL,

category VARCHAR(15) NULL,

quantity INT NULL,

price\_per\_unit FLOAT NULL,

cogs FLOAT NULL,

total\_sale FLOAT NULL

);

**3. Method 1: Direct Import into Retail\_sales Table**

**Step 1: Import CSV directly via SQL Server Import Wizard**

* **Issue:** During import, data types conflicted.
  + Date/time columns imported as **VARCHAR**.
  + **Null/blank** CSV cells **converted to 0 automatically**.
* **Append** option was **disabled** (greyed out).
* Attempting to rename or alter columns gave errors:

**Errors encountered**

* Invalid column names when querying new columns.
* sp\_rename failed with error:

Msg 15248, Level 11, State 1, Procedure sp\_rename, Line 253

Either the parameter @objname is ambiguous or the claimed @objtype (COLUMN) is wrong.

* Imported 0 instead of NULL for blank fields.

**Step 2: Add new columns with correct data types**

ALTER TABLE dbo.Retail\_sales

ADD sales\_date\_new DATE NULL,

sales\_time\_new TIME NULL;

**Step 3: Update new columns by converting existing varchar columns**

UPDATE dbo.Retail\_sales

SET sales\_date\_new = TRY\_CAST(sales\_date AS DATE),

sales\_time\_new = TRY\_CAST(sales\_time AS TIME);

**Step 4: Drop old varchar date/time columns**

ALTER TABLE dbo.Retail\_sales

DROP COLUMN sales\_date, sales\_time;

**Step 5: Rename new columns to original names**

EXEC sp\_rename 'dbo.Retail\_sales.sales\_date\_new', 'sales\_date', 'COLUMN';

EXEC sp\_rename 'dbo.Retail\_sales.sales\_time\_new', 'sales\_time', 'COLUMN';

*Note: If sp\_rename errors occur, double-check column names and spelling.*

**Step 6: Fix NULL values for numeric columns (convert zeros to NULL)**

UPDATE dbo.Retail\_sales

SET customer\_id = NULLIF(customer\_id, 0),

age = NULLIF(age, 0),

quantity = NULLIF(quantity, 0),

price\_per\_unit = NULLIF(price\_per\_unit, 0),

cogs = NULLIF(cogs, 0),

total\_sale = NULLIF(total\_sale, 0)

WHERE customer\_id = 0 OR age = 0 OR quantity = 0 OR price\_per\_unit = 0 OR cogs = 0 OR total\_sale = 0;

**Summary of this method**

* Direct import is quick but prone to type conflicts and NULL mishandling.
* Requires manual schema fixing and data cleanup post-import.
* Risk of data errors and requires careful troubleshooting.

**4. Method 2: Recommended - Use Staging Table**

**Step 1: Create a staging table with relaxed types (all varchar)**

CREATE TABLE dbo.Retail\_sales\_staging (

transactions\_id VARCHAR(100) NULL,

sales\_date VARCHAR(100) NULL,

sales\_time VARCHAR(100) NULL,

customer\_id VARCHAR(100) NULL,

gender VARCHAR(100) NULL,

age VARCHAR(100) NULL,

category VARCHAR(100) NULL,

quantity VARCHAR(100) NULL,

price\_per\_unit VARCHAR(100) NULL,

cogs VARCHAR(100) NULL,

total\_sale VARCHAR(100) NULL

);

**Step 2: Import CSV into Retail\_sales\_staging**

* Use Import Wizard targeting staging table.
* No type conflicts occur due to VARCHAR columns.
* Blank/null values import as empty strings, not zero.

**Step 3: Insert into production table with type conversion and NULL handling**

INSERT INTO dbo.Retail\_sales (

transactions\_id,

sales\_date,

sales\_time,

customer\_id,

gender,

age,

category,

quantity,

price\_per\_unit,

cogs,

total\_sale

)

SELECT

transactions\_id,

TRY\_CAST(NULLIF(sales\_date, '') AS DATE),

TRY\_CAST(NULLIF(sales\_time, '') AS TIME),

TRY\_CAST(NULLIF(customer\_id, '') AS INT),

gender,

TRY\_CAST(NULLIF(age, '') AS INT),

category,

TRY\_CAST(NULLIF(quantity, '') AS INT),

TRY\_CAST(NULLIF(price\_per\_unit, '') AS FLOAT),

TRY\_CAST(NULLIF(cogs, '') AS FLOAT),

TRY\_CAST(NULLIF(total\_sale, '') AS FLOAT)

FROM dbo.Retail\_sales\_staging;

* ***NULLIF(value, '')*** *treats empty strings as* ***NULL****.*
* ***TRY\_CAST*** *attempts conversion, returns NULL if it fails.*
* *This* ***ensures NULLs remain NULL****, no zeros forced.*
* *Data types in final table are strictly enforced.*

**Step 4: Optional - Clean up staging table after import**

TRUNCATE TABLE dbo.Retail\_sales\_staging;

**5. Summary of Errors and Fixes**

| **Error Description** | **Cause** | **Fix / Handling** |
| --- | --- | --- |
| Importing date/time columns as varchar | Incorrect import data type mapping | Add new typed columns, convert, then rename |
| NULL/blank CSV cells imported as zero | Default import behavior | Use NULLIF in UPDATE or insert statements |
| sp\_rename ambiguous column name error | Incorrect syntax or column not found | Verify column names, schema and use fully qualified names |
| Append rows option greyed out | Table schema mismatch or wrong wizard settings | Use staging or fix schema before import |
| Conversion errors on import (e.g., varchar to int) | Data type mismatch | Use staging with all varchar columns |

**6. Recommendation**

**Method 2 (Using Staging Table)** is the most robust, scalable, and maintainable way to import CSV data:

* Avoids import-time data type conflicts.
* Enables explicit, safe data cleansing and transformation.
* Ensures NULL values are preserved correctly.
* Facilitates error handling and repeatable imports.

**7. Final Notes**

* Always verify data types and NULL handling before importing.
* For large and critical data loads, always use a staging table approach.
* Direct imports may be used for quick, non-critical loads but expect to do post-import fixes.
* Use SQL scripts to automate the conversion and cleaning process.